CS 726: Homework #3

Posted: 02/26/2020, due: 03/09/2020 by 5pm on Canvas

Please typeset or write your solutions neatly! If we cannot read it, we cannot grade it.

Note: You can use the results we have proved in class – no need to prove them again.

Q 1. Exercise 4 in Chapter 7 of Recht-Wright (on Canvas). [15pts]

Q 2. Consider the unconstrained optimization problem $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, where f is an L-smooth convex function. Assume that $\|\mathbf{x}_0-\mathbf{x}^*\|_2 \leq R$, for some $R \in (0,\infty)$, and let $f_\epsilon(\mathbf{x}) = f(\mathbf{x}) + \frac{\epsilon}{2R^2}\|\mathbf{x}-\mathbf{x}_0\|_2^2$. Let $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$ and $\mathbf{x}_\epsilon^* = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} f_\epsilon(\mathbf{x})$. You have already shown in previous homework (with possibly minor modifications) that:

$$(\forall \mathbf{x}\in\mathbb{R}^n) : f(\mathbf{x}) - f(\mathbf{x}^*) \leq f_\epsilon(\mathbf{x}) - f_\epsilon(\mathbf{x}_\epsilon^*) + \frac{\epsilon}{2}.$$

(i) Prove that Nesterov's method for smooth and strongly convex minimization applied to f_ϵ will find a solution $\mathbf{x}_k$ with $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$ in $O(\sqrt{\frac{L}{\epsilon}} R \log(\frac{LR^2}{\epsilon}))$ iterations. [5pts]

(ii) Using the lower bound for smooth minimization we have proved in class, prove the following lower bound for L-smooth and m-strongly convex optimization: any method satisfying the same assumption as we used in class (that $\mathbf{x}_k \in \mathbf{x}_0 + \mathrm{Lin}\{\nabla f(\mathbf{x}_0), \ldots, \nabla f(\mathbf{x}_{k-1})\}$) must take at least $k = \Omega(\sqrt{\frac{L}{m}})$ iterations in the worst case to construct a point $\mathbf{x}_k$ such that $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$, for any $\epsilon > 0$. [10pts]

Coding Assignment

Note: Your code needs to compile without any errors to receive any points.

If you are using Python, please follow these rules:

- Please use Python 3.7+.
- You may only use modules from the Python standard library plus NumPy and Matplotlib. Using other third party modules may result in points being deducted.
- You may submit your code in Jupyter notebooks or in .py files. Please archive your files containing your code into one zip/rar/tar file for submission.
- Please include a README file describing how to run your code - if we cannot figure out how to run your code within a reasonable time, you will receive zero points for the entire question.

For the coding assignment, in addition to the methods you have implemented in the last homework, you should also implement the following methods:

- The method of conjugate gradients, for the version we have covered in class (also in Nesterov's book, in Section 1.3.2). You should use the Dai-Yuan rule for β_k (the same as the one we have derived in class).
- The Heavy-Ball method, which applies to L-smooth an m-strongly convex functions, and whose updates are defined by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_1 \nabla f(\mathbf{x}_k) + \alpha_2(\mathbf{x}_k - \mathbf{x}_{k-1}),$$

  where $\alpha_1 = \frac{4}{(\sqrt{L}+\sqrt{m})^2}$ and $\alpha_2 = \left(\frac{\sqrt{L}-\sqrt{m}}{\sqrt{L}+\sqrt{m}}\right)^2$.
- Nesterov's method for smooth and strongly convex optimization (any variant you like – the one we saw in class, or the one from Chapter 4 in Recht-Wright).

Q 3. The problem instance we will consider first is $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, where $n = 100$ and $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{M}\mathbf{x} - \mathbf{b}^T\mathbf{x} + \frac{m}{2}\|\mathbf{x}\|_2^2$ is characterized by:

$$\mathbf{M} = \begin{bmatrix} 2 & -1 & 0 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -1 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

$\mathbf{M}$ and $\mathbf{b}$ can be generated in Matlab using:

```
k = n;
M = diag(2*[ones(k, 1); zeros(n-k, 1)], 0)...
    + diag([-ones(k-1, 1); zeros(n-k, 1)], -1)...
    + diag([-ones(k-1, 1); zeros(n-k, 1)], 1);
b = zeros(n, 1);
b(1) = b(1) + 1;
```

Write the code that implements Nesterov's method for smooth minimization (feel free to reuse the code from last homework), Nesterov's method for smooth and strongly convex minimization, the method of conjugate gradients, and the heavy ball method. Your code should produce three plots, corresponding to three different values of m: (1) $m = 1$, (2) $m = 0.1$, and (3) $m = 0.01$. Each plot should show the optimality gap (on a logarithmic scale) against the iteration count. Each run should be for 1000 iterations. The initial point for all the methods should be $\mathbf{x}_0 = \mathbf{0}$. Discuss your results. Do you observe what you expect from what we saw in class? How does the heavy ball compare to other methods? How about the two variants of Nesterov's method? [20pts]

Now, modify Nesterov's method for smooth minimization so that the function value over the iterates is monotonically decreasing (we have discussed in class how to do that). Ensure that your modified method in each iteration decreases the function value by at least as much as the standard gradient descent (with step size $1/L$; explain how to achieve this). In how many iterations can your new method produce a point $\mathbf{x}_k$ with $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$? (You should provide a theoretical bound.) Produce the same set of plots. Observe what has changed and explain why. [20pts]

Q 4. In this part, you will compare the heavy ball method to Nesterov's method for smooth and strongly convex optimization. Your problem instance is the following one-dimensional instance: $\min_{x\in\mathbb{R}} f(x)$, where

$$f(x) = \begin{cases} \frac{25}{2}x^2, & \text{if } x < 1 \\ \frac{1}{2}x^2 + 24x - 12, & \text{if } 1 \leq x < 2 \\ \frac{25}{2}x^2 - 24x + 36, & \text{if } x \geq 2. \end{cases}$$

Prove that f is m-strongly convex and L-smooth with $m = 1$ and $L = 25$. What is the global minimizer of f? (Justify your answer.)

Run Nesterov's method and the heavy-ball method, starting from $x_0 = 3.3$. Plot the optimality gap of Nesterov's method and the heavy ball method over 100 iterations. What do you observe? What does this plot tell you? [30pts]

Extra Credit Question

Q 5. Suppose that I give you an algorithm (let's call it AGD-G) that given an initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and gradient access to an L-smooth function $f : \mathbb{R}^n \to \mathbb{R}$ (where $0 < L < \infty$) after k iterations returns a point $\mathbf{x}_k \in \mathbb{R}^n$ that satisfies:

$$\|\nabla f(\mathbf{x}_k)\|_2 \leq \sqrt{\frac{2L(f(\mathbf{x}_0) - f(\mathbf{x}^*))}{(k+1)^2}}.$$

Note that AGD-G does not need to know the value of L.

Show that you can use AGD-G to obtain an algorithm that for any m-strongly convex and L-smooth function and any $\epsilon > 0$ can construct a point $\mathbf{x}_k$ with $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$ in $k = O\Big(\sqrt{\frac{L}{m}}\log\big(\frac{L\|\mathbf{x}_0 - \mathbf{x}^*\|_2^2}{\epsilon}\big)\Big)$ iterations. Your algorithm should work without the knowledge of the values of L and m. [15pts]